JOURNAL OF AEROSPACE INFORMATION SYSTEMS
Vol. 10, No. 10, October 2013

Discovering Anomalous Aviation Safety Events Using Scalable Data Mining Algorithms

Bryan Matthews,* Santanu Das,† Kanishka Bhaduri,‡ Kamalika Das,§ Rodney Martin,¶ and Nikunj Oza**
NASA Ames Research Center, Moffett Field, California 94035

DOI: 10.2514/1.I010080

The worldwide civilian aviation system is one of the most complex dynamical systems created. Most modern commercial aircraft have onboard flight data recorders that record several hundred discrete and continuous parameters at approximately 1 Hz for the entire duration of the flight. These data contain information about the flight control systems, actuators, engines, landing gear, avionics, and pilot commands. In this paper, recent advances in the development of a novel knowledge discovery process consisting of a suite of data mining techniques for identifying precursors to aviation safety incidents are discussed. The data mining techniques include scalable multiple-kernel learning for large-scale distributed anomaly detection. A novel multivariate time-series search algorithm is used to search for signatures of discovered anomalies on massive datasets. The process can identify operationally significant events due to environmental, mechanical, and human factors issues in the high-dimensional flight operations quality assurance data. All discovered anomalies are validated by a team of independent domain experts. This novel automated knowledge discovery process is aimed at complementing the state-of-the-art human-generated exceedance-based analysis that fails to discover previously unknown aviation safety incidents. In this paper, the discovery pipeline, the methods used, and some of the significant anomalies detected on real-world commercial aviation data are discussed.

I. Introduction

AS THE complexity and traffic density in the worldwide air transportation system increases, it is going to become more important than ever to develop advanced techniques to help keep the fatal accident rate at the extremely low levels that they are today. A study conducted by The Boeing Company†† in 2012 showed that the fatality rate in that year was less than one per 1 million departures. While this rate is exceedingly small, many aviation experts are concerned that the unprecedented growth in air travel may create a situation where the current safety levels cannot be sustained. This paper demonstrates a new knowledge discovery process, combining novel data mining algorithms developed specifically for detecting precursors to aviation safety incidents that represent a significant advancement in the state of the art.

Current methods rely on human experts to create massive rule-based systems that detect known safety issues, based on whether a small set of parameters exceed some predefined thresholds. This approach, based on known "exceedances," allows analysts to quickly search large databases for predefined issues. While exceedance-based analysis can also offer the ability to detect events while they are happening in an "online" or real-time paradigm, "offline" analysis offers the important ability to allow an analyst to discover something new and explore the data to gain a more holistic understanding of the event. Based on the result of this search, the analyst can recommend new training or operational procedures, maintenance action, or another type of intervention to appropriately address the issue.

The approach that we are discussing here is fundamentally different because, rather than using human-generated rules to detect potential known issues, we use anomaly detection techniques that scan large multivariate time-series databases to uncover statistical anomalies. Our knowledge discovery process outputs a set of statistically significant anomalies from historical observed data. Not all of these anomalies are necessarily significant from the aviation operations perspective. Further input from domain experts such as pilots and airlines safety management teams helps us categorize these anomalies, sometimes based on additional contextual information. Once these unique anomalies are discovered, there is a need to determine the frequency of these occurrences in massive historical time-series databases. We have designed and adapted multivariate time-series search algorithms to efficiently look for high-dimensional anomalies in these massive time-series datasets to address this need. The main contributions to this paper are as follows: demonstrate the integration of novel anomaly detection algorithms into a well-formulated discovery process, show that the process scales with large data, and highlight the types of anomalies that can be identified using this process.

II. Related Work

Anomaly detection is an active area of research in data mining and is widely used in many application areas such as finance, social networks, electronic commerce, earth science, and aeronautics. Broadly speaking, anomalies can be detected using unsupervised, supervised, or semisupervised techniques [1,2]. Unsupervised techniques, as the name suggests, do not require labeled points for detecting anomalies. In this category, the most popular methods are distance-based and density-based techniques, in which anomalies are points in low-density regions or

Received 11 October 2012; revision received 14 May 2013; accepted for publication 1 July 2013; published online 31 October 2013.

*Systems Engineer, Stinger Ghaffarian Technologies Inc.
†Scientist, University Affiliated Research Center.
‡Senior Data Scientist, Mission Critical Technologies, Inc.; currently Netflix Inc., 100 Winchester Circle, Los Gatos, CA 95032.
§Associate Scientist, Stinger Ghaffarian Technologies Inc.; currently University Affiliated Research Center.
¶Computer Engineer. Senior Member AIAA.
**Computer Scientist.
††Data available online at http://www.boeing.com/news/techissues/pdf/statsum.pdf.

points farthest away from other points. In their seminal work, Knorr et al. [3] proposed a distance-based outlier detection technique based on the idea of nearest neighbors. The nave solution has a quadratic time complexity because every data point needs to be compared to every other, in order to find the nearest neighbors. To overcome this, researchers have proposed several techniques, such as the work by Angiulli and Pizzuti [4], Ramaswamy et al. [5], Bay and Schwabacher [6], and Bhaduri et al. [7]. Density-based outlier detection methods, on the other hand, flag a point as an outlier if the point is in a low-density region. Using the ratio of training and test data densities as an outlier score, Hido et al. [8] have proposed a new "inlier"-based outlier detection technique. Supervised techniques require labeled points of both normal and abnormal data for first building a model (e.g., a classifier) and then testing if an unknown data point is an outlier. The model can be probabilistic, based on Bayesian inferencing [9]; or deterministic, such as decision trees, support vector, machines, and neural networks [10]. Very recently, Srivastava [11] used supervised methods for detecting aircraft having abnormal fuel consumption in a fleet commercial aircraft. Semisupervised techniques only require labeled points of normal data. Therefore, they are more widely applicable than the fully supervised ones. These techniques build models of normal data and then flag as anomalies all those points that do not fit the model. The applications we are interested in have very few labeled data points, and we therefore resort to unsupervised outlier detection techniques.

In the context of outlier detection from aviation data, several papers that use support vector machines have been published. Das et al. [12] presented a technique for speeding up one-class support vector machines (SVMs) using a sampling strategy. The authors showed that the proposed technique is 15 times faster than the traditional one-class SVM while maintaining accuracy. Das et al. [13] have also developed an anomaly detection method that can work with both continuous and discrete sequences to demonstrate how some significant anomalies can be detected from real aviation operational data. Autopilot systems allow an aircraft to be flown using the flight computers, and thereby may minimize pilot errors during flight. However, the autopilot systems are controlled using switches in the cockpit that are manipulated by the pilot. SequenceMiner [14] has been shown to detect abnormal switching from a learned model of normal switching. It helps an analyst to identify abnormal pilot inputs used to control the autopilot system. Usefulness of these techniques for aviation safety has been studied in [15], in which the authors have applied these data mining algorithms on real aviation datasets to get meaningful results. It is important to note that the choice of algorithms for anomaly detection in aviation data is based on a number of factors such as the dataset at hand, the types of anomalies we are interested in finding, and the scale of the problem. What is proposed in this paper is a systematic way of combining the strengths of these algorithms in a way that helps an analyst identify new and existing precursors to aviation safety incidents. The algorithms that are part of our system will be discussed in detail in Sec. IV.

The ultimate goal is to develop a process that can complement the existing state-of-the-art human-generated exceedance-based method to uncover some operationally significant events due to environmental, mechanical, and human factors issues in high-dimensional multivariate flight operations quality assurance (FOQA) database. Although there exist a number of algorithms and publications that discuss the use of anomaly detection methods for aviation safety, to the best of our knowledge, this is the first paper that describes the entire knowledge discovery pipeline for aviation data using scalable data mining algorithms. This paper highlights the value of the proposed framework in unearthing safety-critical events in aviation, particularly those that are related to human–automation interaction: a class of events that are becoming increasing important due to the increasing complexity of automation onboard the aircraft. Our framework is the first in aviation analytics that allows automated discovery of such precursors to aviation safety incidents.

III. Background

The world's collection of aircraft and the airspace in which they operate, which we will collectively refer to as "the airspace" from now on, is a complex system that generates a large amount of data and, as such, is a challenging domain for data mining. The airspace contains many elements that are in the critical path of operation, such as aircrafts, airports, people (flight crew), weather events, and routes. Each contains many subcomponents. For example, each aircraft contains many subsystems and components, each of which may have experienced a variety of stresses and maintenance actions. Each airport has multiple runways. These elements that make up the airspace interact in many complex ways. For example, each flight is an aircraft operated by a flight crew to go from one airport to another airport via a route that may be set up or may need to be changed to avoid weather events. Of course, multiple flights operate in the airspace at any given time, and the flights need to coordinate to avoid adverse events (incidents or accidents) while maximizing throughput and minimizing delays.

The airspace provides a very rich and challenging application area for data mining. The complexity described previously manifests itself in a substantial amount and variety of data. There are many data types, only some of which are standard independent, identically distributed data that most methods assume. Data can be temporal, spatial, or spatiotemporal. They can have different sampling rates across different dimensions. Some of it is structured, such as the FOQA data, whereas other kinds of data, such as safety reports and maintenance reports, are unstructured. The data have varying levels of fidelity and accuracy. Methods currently used in the aviation industry to analyze these data are unable to cope with most of this complexity and are typically restricted to simple threshold checks over a small number of variables. While these methods run relatively quickly and are easy to understand, we hypothesize that data mining methods that are more capable of coping with the complexity of airspace data will find operationally significant anomalies that are not revealed by current methods.

We fortunately have an air transportation system that is extremely safe. Even though, ostensibly, the safety and efficiency of the air transportation system seems like a solved problem, there is still room for improvement due to the potential threefold increase in air traffic in the next few decades, as well as room for cost reduction through green aviation measures. Furthermore, adverse events, when they happen, can have devastating consequences. Therefore, improved safety and efficiency based upon the use of various growing data sources remain important objectives. Data mining methods play a critical role in understanding the airspace, not only to correct problems that may be causing "near misses" now but also to anticipate problems that are likely to get worse as the Next Generation Air Transportation System comes about due to increasing air travel, increasing automation in aircraft, and other unforeseen factors. We expect that there are sequences of events hidden in airspace data that lead to or could lead to an adverse event, and we call these sequences precursors. It is critical to identify these precursors so that an on-time detection of such a precursor in flight configuration or flight maneuver can alert the concerned authority to take necessary action to avoid the safety incident in future flights.

IV. Aviation Safety Knowledge Discovery Process

In this section, we describe the entire proposed knowledge discovery process for analyzing aviation data, which we will refer to as the aviation safety knowledge discovery (AVSKD) process. Figure 1 presents a flowchart for this framework. We discuss each block in detail in the following sections.

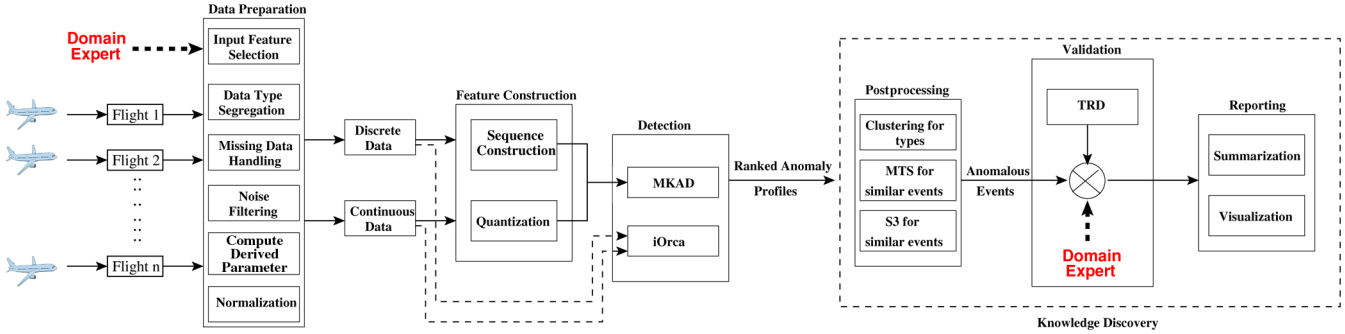


Fig. 1 Aviation safety knowledge discovery process for identifying abnormal aviation events.

A. Data and Preprocessing

1. *Input Data*

Raw FOQA data from each aircraft are collected and input to the AVSKD process. Each flight record is a matrix, with rows corresponding to time samples and columns to observed parameters. The samples are collected at frequency of 1 Hz, and a typical flight from takeoff to landing consists of 5000–6000 samples and 350 parameters. The data are heterogeneous in the sense that there are both discrete (binary and categorical) and continuous parameters.

2. *Data Preparation*

The raw data for each flight are then passed through the data preparation module that performs several functions: feature selection, data type segregation, missing data processing, noise filtering, and normalization. Depending on the type of safety study that is undertaken, aviation experts help us choose the variables (feature selection) that are most relevant for the analysis. To avoid complications due to different aircraft types, all flights for this study were chosen from the same fleet of aircraft. Once the variables are selected, the next subblock separates the discrete and continuous variables so that different data summarization techniques can be applied to them independently. Flight-recorded data are often contaminated with missing data, out-of-bounds variables, noisy recordings, amplitude spikes, etc., caused by sensor malfunctions or recording medium errors. Our next subblock applies several data quality filters specifically developed for this purpose to clean the data. Derived parameters may help to understand the hidden state of the aircraft based on some set of observed parameters, e.g., estimated aircraft speed margin above stall speed based on flap settings, gross weight, and velocity. These parameters have specific advantages for tracking particular events. We construct some derived parameters depending on our study and expert inputs. In the last step of the data preparation phase, we either normalize the continuous features using z score or 0–1 normalization, or we convert for some categorical features (such as flap positions) into a binary representation. The outputs of this block are two time series corresponding to selected discrete and continuous variables for each flight.

3. *Feature Construction*

Once the discrete and continuous parameters are separated, different techniques are applied to the data for feature construction. The continuous data are quantized over a window length and in amplitude to convert them into a symbolic aggregate approximation (SAX)[‡‡] representation [13,16]. The steps taken in the data quality filtering mentioned previously ensure that the features are consistent for an effective conversion to the SAX representation. The discrete parameters are handled by marking the on and off transitions between switch states with unique symbols and concatenating the symbols into a sequence vector while preserving the time ordering. The SAX processing and discrete sequencing is primarily for the multiple-kernel anomaly detection (MKAD) algorithm described in Sec. IV.B.

B. Anomaly Detection Algorithms

Once the data are cleaned and formatted as desired, knowledge discovery algorithms are applied. AVSKD is a very flexible platform, and many anomaly detection algorithms can be easily plugged in. To keep the discussion short, we discuss a couple of algorithms that we have used in our example discovery process.[§§]

1. *Multiple-Kernel Anomaly Detection*

Multiple-kernel anomaly detection is an anomaly detection algorithm designed to run on heterogeneous datasets. Heterogeneity in data may result from the presence of multiple attribute types, e.g., discrete and continuous. MKAD [13] is a "multiple-kernel"-based [17,18] approach, where the major advantage is the method's ability to combine information from multiple data sources. Multiple-kernel learning takes advantage of the mathematics of kernels, allowing users to derive new kernels from existing kernels, provided each kernel satisfies the "Mercer condition" that the kernel function must be continuous, symmetric, and positive definite. In particular, the resultant kernel K can be a convex combination of all kernels computed over multiple features; that is,

$$K(\mathbf{x}_i, \mathbf{x}_j) = \sum_{p=1}^{n} \eta_p \hat{K}_p(\mathbf{x}_i, \mathbf{x}_j)$$

with $\eta_p \geq 0$ and

$$\sum_{i=1}^{n} \eta_p = 1$$

Here, $\hat{K}_p(\mathbf{x}_i, \mathbf{x}_j)$ represents the pth kernel computed for either discrete or continuous parts of data points x_i and x_j, and η_p are to weight the individual kernels [in this paper, we always use $\eta_p = 0.5$ and $n = 2$ (unless otherwise specified), since we did not explore optimal kernel weights

[‡‡]The source code of SAX can be obtained at ~http://www.cs.ucr.edu/~eamonn/SAX.htm [retrieved 2013].
[§§]Some or all these algorithms can be downloaded from https://c3.nasa.gov/dashlink/.

or construct additional kernels in this paper]. There exist several classes of kernel that coincide with the Mercer kernel, such as radial basis function, polynomial, bag of words, sigmoid, spline, graph-based, tree-based, and mismatch-based functions [19–21]. Although the current MKAD approach is not limited to the choice of kernel functions, for the purposes of this study, the normalized longest common subsequences (NLCS)-based [14] kernel function is used to model switching sequences for the process, where the order of the switching is important from an operational perspective. Even though the same similarity function is being used for each kernel, it is important to note that the features have completely different meanings. One feature is the switching of autopilot states, while the other is a discretized version of continuously recorded variables. Since this is the case, the features cannot simply be combined into a single feature vector where the similarity function can be used to form the final kernel; hence, the need for a multiple-kernel approach where the features can be combined in the kernel space. This kernel can be defined as NLCS(X, Y), where X and Y are two sequences of discrete symbols and

$$\mathrm{NLCS}(X, Y) = \frac{|\mathrm{LCS}(X, Y)|}{\sqrt{|X||Y|}} \tag{1}$$

Given two sequences, X and Z, Z is a subsequence of X if removing some symbols from X produces Z. Z is a common subsequence of sequences X and Y if Z is a subsequence of both. The longest such subsequence is called the longest common subsequence (LCS) and is denoted by LCS(X, Y), and $|\mathrm{LCS}(X, Y)|$ is its length. This kernel is used because there exist standard operating procedures for flying the aircraft, and the sequence of the pilot inputs (or actions) along with the measured quantities or parameters are extremely meaningful. NLCS is a useful metric for measuring similarity between discrete sequences. These sequence features can be generated directly from discrete parameters or from SAX representations [16] of continuous variables, as described in Das et al. [13].

The heart of MKAD is a one-class SVM model that constructs an optimal hyperplane in the high-dimensional feature space to separate the abnormal (or unseen) patterns from the normal (or frequently seen) ones. This is done by solving the following optimization problem [22]:

$$\min\ Q = \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j K(x_i, x_j) \quad \text{subject to } 0 \le \alpha_i \le \frac{1}{\ell\nu}, \qquad \sum_i \alpha_i = 1, \qquad \rho \ge 0, \qquad 0 < \nu < 1 \tag{2}$$

where α_i are Lagrange multipliers, ℓ is the number of examples, ν is a user-specified parameter that defines the upper bound on the training error (and the lower bound on the fraction of training points that are support vectors), ρ is a bias term, and K is the kernel matrix. Once this optimization problem is solved, at least $\nu\ell$ training points with nonzero Lagrangian multipliers (α) are obtained and the points for which $\{x_i\colon\ i \in [\ell], \alpha_i > 0\}$ are called support vectors. Once $\boldsymbol{\alpha}$ is obtained, we can compute the decision function

$$f(\mathbf{z}) = \mathrm{sign}\left(\sum_i \alpha_i \sum_p \eta_p \hat{K}_p(\mathbf{z}, \mathbf{x}_i) - \rho\right)$$

to predict a positive or negative label for a given test vector $\mathbf{z}$. Examples with negative labels are classified as outliers.

MKAD is highly scalable, and it can process large datasets very fast. We conducted a scalability study where we were able to process the data of 940,000 flights (equivalent to over 5 billion data points) in 12.5 h using distributed computing.

2. *Index-Orca*

Another unsupervised anomaly detection method based on the distance to nearest neighbors as the measure of "outlierness" [7,6] is also used in this framework. For this method, the raw time-series data are presented as features to the algorithm after the appropriate data quality filters have been implemented. The basic idea of distance-based outlier detection is to find, for each point x in the dataset, k other points that are closest to x by computing the Euclidean distance between x and all the other points in the dataset. Then, the points are ranked (from highest to lowest) according to the average distance (or maximum distance) to the k-nearest neighbors. This average distance is known as the outlier ranking function or outlier score. The ranked list becomes the set of outliers in the order specified by the score. It is easy to verify that this computation has a computational complexity of $O(n^2)$, where n is the number of points in the dataset. To speed up this computation, Bay and Schwabacher [6] proposed the state-of-the-art algorithm Orca, which uses a pruning strategy to quickly remove the normal points that are the bulk of the population. To implement pruning, one needs to keep track of the smallest outlier score of the outlier set (called the cutoff c). Then, while testing a new point x, its nearest neighbors are searched, and whenever a neighbor of x is found for which the distance to x is less than c, then x is discarded from the pool of outlier candidates.

More recently, Bhaduri et al. [7] proposed the new method Index-Orca (iOrca), which is at least an order of magnitude faster than Orca while still guaranteeing correct results. The inefficiency of Orca is due to its slow update of cutoff (hence pruning). To overcome this, [7] proposes a data reordering scheme such that 1) the cutoff is updated faster, and 2) for every point, the nearest neighbors are found quickly, often in constant time.

To update the cutoff faster, iOrca processes the outliers as early as possible. To achieve this, a random point R is first selected as a reference point from D, and then the distance of all the other points in D from R is calculated. This array of distances is the index, which is then used to reorder the points in D with increasing distance from R. Instead of traversing through the data in the original order (or random order [6]), iOrca tests the points along this index order. The rationale for this traversal is as follows. Since R is chosen at random, it is likely that R will be one of the inlier points if there are more inliers than outliers. Therefore, the points farthest from R will be more likely be the outliers. Since the points in D are processed in decreasing distance to R, it is very likely that the outliers will be processed first, leading to a faster increase in the cutoff threshold. Reordering the database for testing also allows iOrca to terminate without ever needing to process all the data:

Lemma IV.1 [stopping rule]: Let L be the index as described in this section. Let R be the reference point used to build L. Let x_t be any test point currently being tested by iOrca. If

$$\|x_t - R\| + \|R - x_{n-k+1}\| < c$$

then iOrca can terminate the execution immediately, where c is the current cutoff threshold, x_{n-k+1} is the true kth nearest neighbor of R, and $\|\cdot\|$ is the 2-norm of a vector.

Finally, the nearest neighbors of a test point are searched along the index order. Instead of starting the search from the beginning of the database for every test point, iOrca starts the search from the location upon which it lies along the index, and then it gradually "spirals" out. Since the index

prescribes a total ordering of the points projected along one dimension, it is likely that, for a test point x_t, if x_i is closer to x_t than x_j along the index, then it will be the same even when the actual distances are also computed, i.e., if

$$|\|x_i - R\| - \|x_t - R\|| < |\|x_j - R\| - \|x_t - R\||$$

then it is expected that

$$\|x_t - x_i\| < \|x_t - x_j\|$$

As shown in Bhaduri et al. [7], iOrca is highly scalable and often produces a near-linear running time due to the index and early stopping criterion.

Both algorithms discover anomalies. They also decompose the overall anomaly score to reflect the contribution of each parameter toward that score, thus providing a rough estimate of the most faulty parameter. These two methods previously described are not intended to solve the same problem but rather to detect anomalies at different levels of abstraction. For instance, one important distinction between MKAD and iOrca is that MKAD works with the data compressed to single similarity comparisons for each flight (using SAX, sequencing, and the $_{\text{NLCS}}$ kernel), while iOrca ingests all the raw time-series data logged from the flight data recorder and detects anomalies at each time point. Since MKAD is working with the data at a lower resolution, it reports anomalies at the fleet level, i.e., it identifies anomalous flights, whereas iOrca is more tuned to detect anomalies within a flight because it analyzes the uncompressed time series directly.

C. Knowledge Discovery

1. *Postprocessing*

Both of these outlier detection algorithms output anomaly scores and the location of the anomalies. Depending on the type of algorithm used, we can either identify an entire flight as abnormal (e.g., MKAD) or point out at which point in time the abnormality has been found (e.g., iOrca). Depending on the size of the input database, the algorithms may discover hundreds to thousands of ranked anomalies, making it difficult for domain experts to validate all of them. We have adopted a variant of the *k*-means algorithm that helps us to cluster the found anomalies based on the contribution score of each parameter. Flights that share the same weighting on each of the anomalous parameters will be clustered together, and therefore have similar anomalous flight characteristics. Then, instead of presenting all the anomalies to the domain expert, we only show them some representative examples (such as the medoid flight) for each group.

Once anomalies of interest are discovered using any of these anomaly detection techniques, we are interested in knowing the frequency and severity of those events in the entire dataset. Due to the statistical nature of the algorithms, not all events that have similar characteristics or hold operational significance are guaranteed to be grouped together in the anomaly rankings. This raises the need for a tool to search for similar events in the remaining dataset that a domain expert has identified as operationally significant from the list of anomalies. To do this, we use a tool called Multivariate Time-Series Search (MTS) [23]. For each anomaly found earlier, this tool searches for similar anomaly evidences in the entire database. The MTS tool takes as input the data files (which can be very large), a query in the form of a multivariate time series defined over a small subset of variables, and a threshold ϵ specifying the radius of the search with respect to the query. It returns all the location pairs in the form of (file number, time instance) where a match has been found. In the preliminary phase, an index is built on the dataset. This consists of selecting a random point from the dataset (called the reference point) and then rearranging all the other points in the dataset according to distance to this fixed point. This process is repeated separately for all the variables in the dataset, even before the query is provided. When the query q arrives in the form of a multivariate waveform, first it is mapped to the "index space" by subtracting the reference point, and then a binary search is executed to find the location of the transformed query q' on the index. Using triangle inequality, it is easy to show that the only candidates of interest are the tuples in the range $q' \pm \epsilon$, which guarantees no false dismissals. These candidates are then fetched from the dataset, and false positives are removed by doing an exact calculation with q.

Similarly, for anomalies in discrete sequences, such as those found by MKAD with contributions from the discrete kernel, we pass the examples to our tool sequence similarity search (S3). All examples are converted into sequences using the preprocessing step described earlier. The query sequence is then compared against all sequences in the data using the NLCS metric as the distance function described in Eq. (1). The distances are sorted, and a user-defined cutoff is applied to determine the matches.

2. *Validation*

Once we find anomalous events using the knowledge discovery algorithms described previously, we validate them. There are three sources of validation that are generally followed in the aviation industry. The first level of validation is with the help of experts in the field. For our case study, we have sought help from four aviation experts with differing backgrounds. The first domain expert has over 30 years of experience in human factors and aviation, text analysis, and statistical methodologies. The second expert is a retired commercial airline pilot of a major U.S. carrier with over 35 years of experience in flying Boeing 777 and 747 aircrafts. Our third expert is a researcher in human factors, with special emphasis on aircraft automation and its effects on human performance. He is also a commercial pilot and flight instructor for single- and multiengine aircrafts, including the Airbus aircraft. Our fourth domain expert is also a researcher in the human factors, with emphasis in cognitive science and psychology as applicable to spatial reasoning, decision making, risk assessment, communication, and skill acquisition and retention for air traffic controllers, airline pilots, space mission controllers, and astronauts. He has over 35 years of experience in flying single-engine and multiengine commercial aircraft, including certifications for Airbus A320 and A330, and Boeing B737.

As further validation, a repository of text reports, known as the text reports database (TRD), written by a member of the flight crew can potentially be used to confirm the anomaly with the pilots depiction of the event. While the numeric FOQA data give us information as to what happened, the text reports, if available, provide information as to why the event happened. A third way of validation is by interviewing the crew themselves once an event is found. However, in many cases, we do not have access to either the reports database or the crew, and so we mainly rely on experts in the field for validation.

3. *Reporting System*

The final step in the AVSKD process is the report generation phase. We have developed a web-browser-compatible reporting system,[¶¶] which displays each anomalous flight and the top few parameters to which it is deemed anomalous. For a given anomalous flight identifier, it reports the following: 1) percentage contribution (discrete, continuous), graphical plot (contributing continuous parameters), 3) percentage contribution of each continuous parameter, and 4) missing and extra discrete switches.

[¶¶]A sample of the report can be downloaded from https://c3.nasa.gov/dashlink/static/media/other/html_report.png.

Table 1 Automated continuous and discrete parameters[a]

Modes	Parameters
Continuous	Altitude, airspeed, roll, pitch, angle of attack, engine, RPM, windspeed, computed airspeed above stall, aileron, rudder, stabilizer, elevator, etc.
Discrete	Landing gear, landing flap position, autopilot and/or flight director status various vertical and lateral guidance system modes

[a]For further explanation of automated modes, please refer to the Airbus/Boeing flight crew training manuals at http://www.737ng.co.uk/docs.htm [retrieved 2013].

V. Discoveries

This section will discuss three examples of anomalies found by each of the detection algorithms. It is important to note that the choice of the examples presented in this paper is based on the fact that both the domain experts and airline agreed that they are compelling incidents. Since this is a discovery problem on unlabeled data, the results can be validated for ground truth only by domain experts. It can be argued that the current threshold-based methods can be used to look at false-positive and false-negative rates to generate receiver operating characteristic curves, and therefore tune the algorithm's performance. However, this approach will only affect the algorithm's ability to detect what is currently being detected and not improve the algorithm's ability to detect unknown operationally significant events. Two of the examples identified are due to pilot-controlled maneuvers (see Sec. I). The other example involves a human–machine interaction scenario that is a significant challenge for the pilots to maintain awareness of the modes of the flight computer (see Sec. II). All three examples are important safety events. They appeared within the top-ranked anomalies; however, the rank is determined by the statistical severity of the identified anomalous flights. In other words, anomalies with operational significance may not always end up on the top of the list. In FOQA data, there are several natural sources of homogeneity; for example, flights that have common origin or destination airports, city pair routes, tail numbers, or aircraft models, as well as seasonal aspects (such as flights within a month) can be grouped accordingly. Even within a flight, there exist several phases, such as takeoff, cruise, and landing. In this study, we partition the data based on departure or destination airport and analyze the takeoff and landing, respectively. The set of continuous parameters considered include altitude, airspeed, roll, pitch, engine revolutions per minute (RPM), windspeed, estimated airspeed above stall, and various control surface positions. The binary state parameters considered include landing gear, landing flap position, whether the autopilot and/or flight director was engaged, and various vertical and lateral guidance system modes (see Table 1).

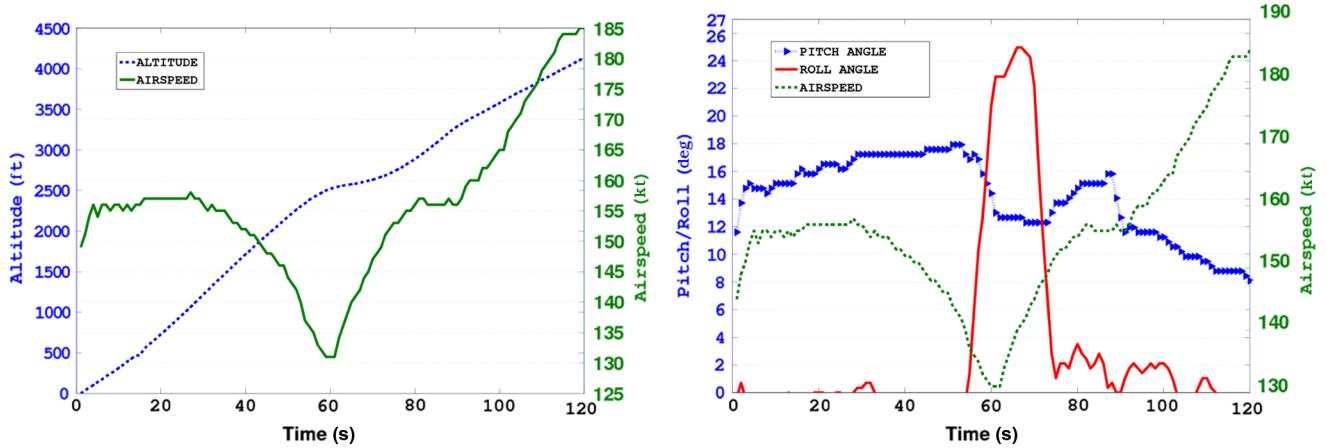
A. Fleet Level Anomalies

The following two anomalies were identified at the fleet level, meaning individual flights were labeled anomalous by the algorithm and not the samples within the flight. Further analysis was performed in the postprocessing block to look within the flight to determine what characteristics of the flight were abnormal.

1. *Drop in Airspeed*

An anomaly was found in a flight by MKAD, using both the continuous and the discrete parameters listed at the beginning of Sec. V as input, and formatted as described in Sec. IV.B. The algorithm was run on all available flights departing from a particular airport, identifying this flight as an anomaly, which was ranked 11th out of 439 flights, with a runtime of 3 s. The higher-ranked anomalies were examined and found to have various statistically anomalous characteristics (such as lower airspeeds, high windspeeds, or an uncommon departure direction); however, the behavior in this flight was deemed to be of elevated interest by the domain experts.

Takeoff procedures typically allow aircraft to maintain an airspeed of $V_2 + 10$ kt, where V_2 is the minimum speed that needs to be maintained up to acceleration altitude. Acceleration altitude is the altitude [typically 3000 ft above ground level (AGL)] at which pitch is reduced slightly and positive climb maintained, allowing acceleration through flap retraction speeds to 250 kt. In the flight shown in Fig. 2a, approximately 30 s after takeoff, the aircraft began to experience a drop in airspeed. Given the gross weight and flap settings on the aircraft at the time, the aircraft decelerated to only 12 kt above the estimated stall speed. The experts agreed that this flight was of interest, since it is highly unusual at this point in the flight to experience such a drop in airspeed. This drop in airspeed was a direct result of the pitch angle being held too high for too long, under the conditions that both engines were running at maximum climb thrust and the autopilot was not yet engaged. The drop in airspeed continued for another 30 s. At that time, a turn maneuver was executed with a simultaneous correction of the pitch angle (see Fig. 2b), allowing the aircraft to start increasing airspeed again.



a) Drop in airspeed with the corresponding altitude at which it occurred

b) Airspeed recovery as the pilot performed a roll maneuver and simultaneously began correcting the pitch to begin increasing airspeed

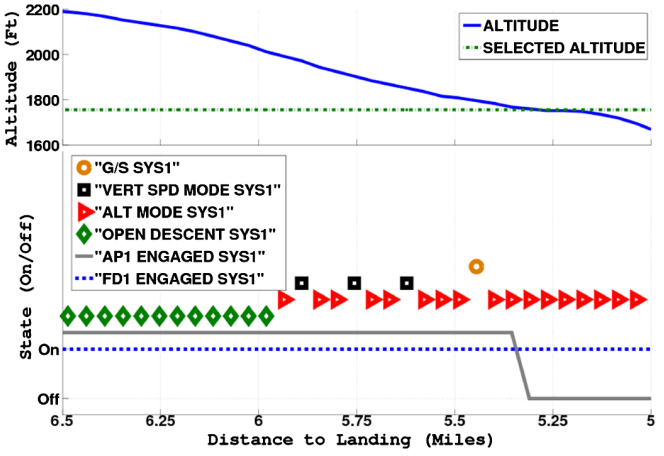
Fig. 2 Drop in airspeed.

To find similar incidents, we ran the MTS search tool on a particular airport that is known to the domain experts to have unusual takeoff procedures. The airport had 9543 flights consisting of roughly 1 million data points. We have used only two parameters for search, altitude and airspeed, and the aforementioned example as the query. Building the index took 21 min. (only takeoff phase). Searches for other similar events took 0.08, 0.09, and 0.1 s for thresholds of 0.5, 1, and 2, returning 4, 17, and 69 hits, respectively. While visual inspection reveals similarity with the query, we are currently validating them with our experts.

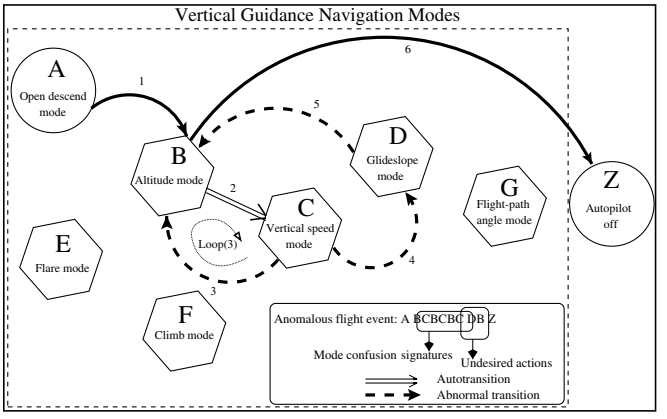
2. *Mode Confusion*

An anomaly was found in a flight by MKAD, using only the discrete parameters listed at the beginning of Sec. V as input and 100% weights on the discrete kernel, and it was formatted using the sequence preprocessing steps described in Sec. IV.B. An additional step was taken to ignore the flight computer modes when both the autopilot and flight director were disengaged, since no action would be taken by the autopilot or pilot in response to these mode states, and therefore they would not be relevant to this study. The algorithm was run on all available flights landing at a particular airport with a runtime of 210 s. This anomaly ranked 13th out of 19,243 flights. Again, the higher-ranked anomalies were investigated and found to be higher ranked due to modes in the data that are infrequently used (such as flight-path angle mode and altitude mode late in approach); however, this particular event was found to be of high interest by the domain experts. The anomaly description follows:

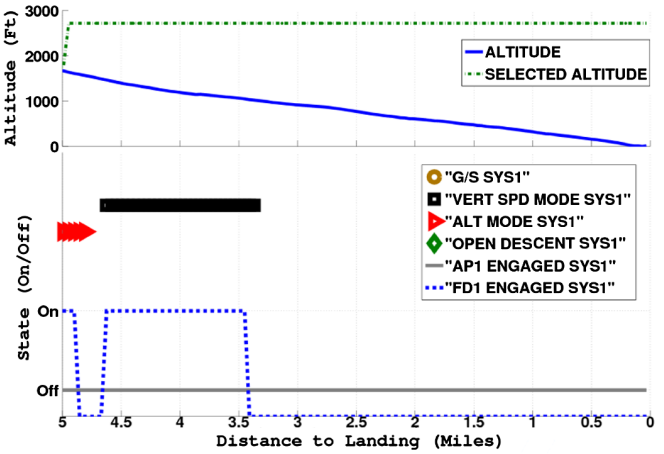
On descent for approach, the aircraft was slightly above, yet on track, to intercept the glideslope (G/S). At approximately 2000 ft AGL, the autopilot vertical guidance system was manually switched from open descent mode into altitude mode. The autopilot immediately reverted to the default vertical speed mode. Manual selection of altitude mode was attempted repeatedly with the same result (see Fig. 3a). The modes listed in Fig. 3a are defined as: G/S SYS1, glide slope vertical mode; VERT SPD MODE SYS1, vertical speed vertical mode; ALT MODE SYS1, altitude hold vertical mode; OPEN DESCENT SYS1, open descent vertical mode; AP1 ENGAGED SYS1, autopilot 1 engaged; FD1 ENGAGED SYS1, flight director 1 engaged. The glideslope mode was engaged momentarily just before the autopilot was manually disengaged, at which time the altitude mode was manually reselected. Figure 3b presents the corpus of vertical guidance modes for this particular flight, and it helps to identify the potential anomalous sequences of switching initiated by the human. The notation for Loop 3 indicates the state sequence was repeated three times. The anomalous mode transitions are shown in dashed lines in Fig. 3b. There are two important observations. First, we see a signature of mode confusion where the operator repeatedly attempted to override mode C with mode B (see Fig. 3b); however, the flight computer did not accept the command and autotransitioned back to mode C. If the aircraft had been closer to the selected altitude, the flight computer would have accepted the command and the unnecessary mode switching could have been avoided. Second, this was followed by a sequence of undesired actions (C → D → B) briefly before the pilot removed himself/herself from the undesired automation by manually disengaging the autopilot resulting in mode *Z*. The domain experts identified these sequences of actions as potential evidence of mode confusion.



a) Evidence of mode confusion with the unusual mode switching

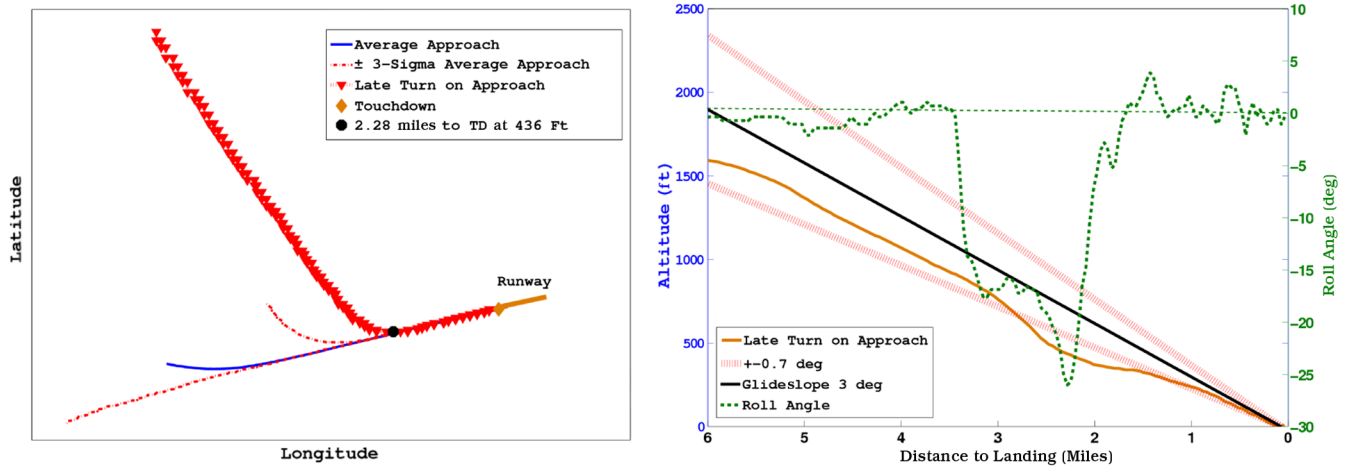


b) Conceptual diagram of the state space and path taken for the vertical guidance system in this example



c) Confirmation of mode confusion with the recycling of the flight director

Fig. 3 Mode confusion (The modes listed in the figure above are defined as: G/S SYS1: glide slope vertical mode, VERT SPD MODE SYS1: vertical speed vertical mode, ALT MODE SYS1: altitude hold vertical mode, OPEN DESCENT SYS1: open descent vertical mode, AP1 ENGAGED SYS1: autopilot 1 engaged, and FD1 ENGAGED SYS1: flight director 1 engaged.).



a) Late turn on approach compared to the average flight track for this runway

b) Glide slope compared to the 3 deg glidepath showing the altitude at which the roll was performed

Fig. 4 Unstable approach.

At this point, the aircraft was hand flown to level off at the selected altitude. A new altitude of 2700 ft AGL was then selected, presumably to program the flight computer for the missed approach maneuver in case of a go around. At this point, the flight director was manually reset, which resulted in the default vertical speed mode being selected (see Fig. 3c). The aircraft was then manually flown using glideslope indicators and by using the flight director in vertical speed mode. At 1000 ft AGL, the flight director was disengaged for the final time, and this was followed by a visual landing. Repeated failed attempts to select the desired vertical modes, subsequent disengagement of the autopilot, and recycling of the flight director all suggest mode confusion, which added unnecessary workload to an already highly demanding situation.

A search for similar examples was performed on the entire dataset using S3. The sequence from the aforementioned example was used as the query and compared against the full set of landing sequences in the data, with a running time of 127 min. One other flight was found with near-identical behavior, while 12 other flights were found to have either the unusual switching behavior or the recycling of the flight director, but not both.

B. Flight Level Anomalies: Unstable Approach

The last anomaly was identified at the flight level, meaning time samples within a flight were labeled anomalous by the algorithm. One advantage in using this approach is that the algorithm can directly identify the abnormal characteristics within the flight, however there is a trade off with runtime on larger datasets.

An anomaly was found in a flight by iOrca, using the set of continuous parameters listed at the beginning of Sec. V as input, and formatted as described in Sec. IV. The top N was chosen to be $1000 \approx 0.2\%$ of the full data, and k neighbors were set to the algorithm's default value of 5. The algorithm was run on all available flights landing at a particular airport with a runtime of 622 s using 519,025 time samples. This flight was identified as an anomaly and ranked 22nd out of 2800 flights. Again, the higher-ranked anomalies were investigated and found to be statistically unusual (higher airspeeds at lower altitudes); however, this particular event was found to be of notable interest by the domain experts. The anomaly description follows:

For this flight, an aircraft was being flown manually during the approach without the flight director, presumably for a visual approach, since the reported weather was clear. The aircraft was being flown slightly under a 3 deg approach path and was on a base leg (the part of the approach that is perpendicular to the runway) (see Fig. 4a for flight track). Late turn on approach compared to the average flight track for this runway. The aircraft made the turn to final at 2.28 miles to touchdown (TD) at 436 ft above ground level (see Fig. 4b). An unusually sharp turn was executed to intercept the final approach path at 2.28 miles. During this turn, the aircraft was below 500 ft AGL, and the turn required a bank angle of 26 deg to intercept the final approach course (see Fig. 4b). Such a turn so close to the airport is undesirable, since it does not allow much time for the aircraft to meet stable criteria for landing, which could result in a go around. However, in this case, correction of the aircraft's attitude allowed for stabilized conditions to be met before touching down safely.

C. Value of Discoveries

The AVSKD process has helped in identifying important precursors to undesirable events. We are defining precursors as the combination of conditions that increase the likelihood of potential unsafe situations in the future. In the aforementioned flight discussed in Sec. V.A.1, the drop in airspeed may have led to a stall warning. Similarly, the mode confusion event discussed in Sec. V.A.2 could have lead to a failure to reach stabilized landing criteria, and the unstable approach in Sec. V.B. might have resulted in a go around or hard landing. Insights derived from the precursors identified by the AVSDK process can be brought to the attention of the airline's training personnel and policy makers, and further understanding of the frequency of these events can provide additional justification to initiate proactive operational changes to address safety-related issues.

The current FOQA program is designed to provide answers to questions that the domain experts have thought to ask, whereas in the AVSKD process, we are not focusing on any particular group of anomalies, instead we "search for the unexpected" from the entire dataset and consult the domain expert to validate each of them. Exceedance-based programs are like mechanical filters that need prior knowledge of the events before they can be implemented. Looking at all of the data using the exceedance-based approach is not scalable, and therefore has been implemented only on the portions of the flights as prescribed by the domain experts, leaving many unexamined portions of the data. In our approach, the algorithms need no prior knowledge of the events and do not examine only a subset of the data. Most of the algorithms presented in this paper have the ability to handle large datasets, and therefore allow us to process the full dataset in a reasonable amount of time. It is also important to note that FOQA programs are dependent on the validity of the thresholds defined by the domain experts, whereas our methods use statistical properties of the data to rank the anomalies and find their severities. If multiple parameters interact in a counterintuitive way, the exceedance-based approach might overlook a subset of those parameters if they are not predefined by the domain experts, whereas our multivariate models can take into account the complex relationships and report their contributions to the anomalies.

VI. Conclusions

In this paper, a data mining and knowledge discovery process has been described that can detect precursors to aviation safety incidents. This process was demonstrated on a large commercial aviation dataset by showing that it identified three operationally significant events that were validated by experts. It is planned to extend this work to identify anomalous sequences across a set of flights that have a key characteristic in common, such as the same aircraft, flight crew, or city pair. Work is also being done to identify and incorporate other sources of data that correspond to these flights, such as radar track data and weather data, as these data will also be useful in identifying anomalies. Making this framework even more scalable to facilitate larger-scale analysis is also currently being worked on. Finally, it should be noted that, although the methods described are for the aviation domain, all these techniques are domain agnostic, and they can be used in any application area that exhibits similar data properties and challenges. For example, in recommender systems, the task it to maximize the click-through rate by good quality recommendations based on models learned from user browsing data. These models are learned on continuous data, and the discrete signals may encode seasonal patterns that help in model segmentation and selecting the appropriate model for predicting user behavior. Anomaly detection in this case relates to finding unique, noisy, or fraudulent users for further analysis or building better models.

Acknowledgments

This research is supported by the NASA Aviation Safety Program's System-Wide Safety and Assurance Technologies Project. We would like to thank Ashok Srivastava and John Stutz for their contributions and feedback. We would also like to thank Irv Statler, Bob Lawrence, Mike Feary, Immanuel Barshi, and the partner air carrier for providing data and expertise.

References

[1] Chandola, V., Banerjee, A., and Kumar, V., "Anomaly Detection: A Survey," *ACM Computing Surveys*, Vol. 41, No. 3, 2009, pp. 1–58. doi:10.1145/1541880

[2] Hodge, V., and Austin, J., "A Survey of Outlier Detection Methodologies," *Artificial Intelligence Review*, Vol. 22, No. 2, 2004, pp. 85–126. doi:10.1023/B:AIRE.0000045502.10941.a9

[3] Knorr, E. M., Ng, R. T., and Tucakov, V., "Distance-Based Outliers: Algorithms and Applications," *VLDB Journal*, Vol. 8, Nos. 3–4, 2000, pp. 237–253. doi:10.1007/s007780050006

[4] Angiulli, F., and Pizzuti, C., "Outlier Mining in Large High-Dimensional Data Sets," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 17, No. 2, 2005, pp. 203–215. doi:10.1109/TKDE.2005.31

[5] Ramaswamy, S., Rastogi, R., and Shim, K., "Efficient Algorithms for Mining Outliers from Large Data Sets," *ACM SIGMOD Record*, Vol. 29, No. 2, 2000, pp. 427–438. doi:10.1145/335191.335437

[6] Bay, S. D., and Schwabacher, M., "Mining Distance-Based Outliers in Near Linear Time with Randomization and a Simple Pruning Rule," *KDD'03: Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM, New York, NY, 2003, pp. 29–38.

[7] Bhaduri, K., Matthews, B. L., and Giannella, C. R., "Algorithms for Speeding Up Distance-Based Outlier Detection," *KDD '11: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM, New York, NY, 2011, pp. 859–867.

[8] Hido, S., Tsuboi, Y., Kashima, H., Sugiyama, M., and Kanamori, T., "Inlier-Based Outlier Detection via Direct Density Ratio Estimation," *ICDM'08: Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, 2008, pp. 223–232.

[9] Das, K., and Schneider, J., "Detecting Anomalous Records in Categorical Datasets," *KDD'07: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM, New York, NY, 2007, pp. 220–229.

[10] Hu, W., Liao, Y., and Vemuri, V. R., "Robust Anomaly Detection Using Support Vector Machines," *ICMLA'03: Proceedings of the International Conference on Machine Learning and Applications*, Morgan Kaufmann Publishers Inc., San Francisco, CA, 2003, pp. 168–174.

[11] Srivastava, A. N., "Greener Aviation with Virtual Sensors: A Case Study," *Data Mining and Knowledge Discovery*, Vol. 24, No. 2, 2012, pp. 443–471. doi:10.1007/s10618-011-0240-z

[12] Das, S., Bhaduri, K., Oza, N. C., and Srivastava, A. N., "ν-Anomica: A Fast Support Vector Based Novelty Detection Technique," *IEEE Conference of Data Mining*, IEEE, Piscataway, NJ, 2009, pp. 101–109.

[13] Das, S., Matthews, B. L., Srivastava, A. N., and Oza, N. C., "Multiple Kernel Learning for Heterogeneous Anomaly Detection: Algorithm and Aviation Safety Case Study," *KDD '10: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM, New York, NY, 2010, pp. 47–56.

[14] Budalakoti, S., Srivastava, A., and Otey, M., "Anomaly Detection and Diagnosis Algorithms for Discrete Symbol Sequences with Applications to Airline Safety," *IEEE Transactions on Systems, Man, and Cybernetic, Part C: Applications and Reviews*, Vol. 39, No. 1, 2009, pp. 101–113. doi:10.1109/TSMCC.2008.2007248

[15] Das, S., Matthews, B. L., and Lawrence, R., "Fleet Level Anomaly Detection of Aviation Safety Data," *IEEE Conference*, Prognostics and Health Management, Piscataway, NJ, June 2011, pp. 1, 10, 20–23.

[16] Lin, J., Keogh, E. J., Wei, L., and Lonardi, S., "Experiencing Sax: A Novel Symbolic Representation of Time Series," *Data Mining and Knowledge Discovery*, Vol. 15, No. 2, 2007, pp. 107–144. doi:10.1007/s10618-007-0064-z

[17] Bach, F., Lanckriet, R. G., and Jordan, M., "Multiple Kernel Learning, Conic Duality, and the SMO Algorithm," *Proceedings of the Twenty-First International Conference on Machine Learning*, ACM, New York, NY, 2004, p. 6.

[18] Lanckriet, R. G., Cristianini, N., Bartlett, P., Ghaoui, L. E., and Jordan, M. I., "Learning the Kernel Matrix with Semidefinite Programming," *Journal of Machine Learning Research*, Vol. 5, Dec. 2004, pp. 27–72.

[19] Schölkopf, B., and Smola, A., *Learning with Kernels*, MIT Press, Cambridge, MA, 2002, pp. 25–55.

[20] Kashima, H., Tsuda, K., and Inokuchi, A., "Kernels for Graphs," *Kernel Methods in Computational Biology*, Vol. 39, No. 1, 2004, pp. 101–113.

[21] Chapelle, O., and Haffner, P., "Support Vector Machines for Histogram-Based Classification," *IEEE Transactions on Neural Networks*, Vol. 10, No. 5, 1999, pp. 1055–1064. doi:10.1109/72.788646

[22] Schölkopf, B., Platt, J. C., Shawe-Taylor, J. C., Smola, A. J., and Williamson, R. C., "Estimating the Support of a High-Dimensional Distribution," *Neural Computation*, Vol. 13, No. 7, 2001, pp. 1443–1471. doi:10.1162/089976601750264965

[23] Bhaduri, K., Zhu, Q., Oza, N. C., and Srivastava, A. N., "Fast and Flexible Multivariate Time Series Subsequence Search," *ICDM '10: Proceedings of the 2010 IEEE International Conference on Data Mining*, IEEE Computer Society, Washington, D.C., 2010, pp. 48–57.

A. Banerjee
Associate Editor